NSF Compiler Workshop
September 5, 2001

Jack Davidson

Department of Computer Science

University of Virginia

Two research opportunities

- **Compilation for *high-performance* embedded systems**
- **Compilation for heterogeneous, networked computers-Internet computing**

Compilation for Embedded Systems

- **Why is this important?**
  - **Embedded systems are key building blocks of vital national infrastructure**
  - **Embedded processors are key components in a wide array of consumer devices**

Compilation for Embedded Systems

- **What's the problem?**
  - **Mainframe/desktop compilation paradigm is inadequate for embedded systems**
- **Result**
  - **Embedded software is still written in assembly language**
    - **Higher development and maintenance costs**
    - **Slower time to market**
    - **Hinders innovation**

Compilation for Embedded Systems

- **Why is compilation for embedded systems hard(er)?**
  - **Many processor variants each with special features**
  - **Cross-cutting constraints of speed, power, and size**
  - **Very performance and cost sensitive**
  - **Custom and semi-custom processors**

Compilation for Networked Systems

- **Why is this important?**
  - **Duh!!**
  - **Internet computing**
  - **Ubiquitious computing (clusters, motes, swarms, hives, herds)**

Compilation for Networked Systems

- **What's the problem?**
  - **Mainframe/desktop compilation paradigm is inadequate for networked systems**
- **Result**
  - **The potential of networked systems has not been fully realized**

Compilation for Networked Systems

- **Why is compilation for networked systems hard(er)?**
  - **Heterogeneous platforms**
  - **Dynamic environment (changing QoS and resources)**
  - **Code is injected dynamically from different sources**
  - **Often working at the binary level**
  - **Continuous operation**

Meeting these challenges

- **Move from static to more dynamic compilation approaches**

Meeting these challenges

- **Embedded systems: new compilation framework**
  - **Adaptive compilation-compiler configured at compile-time for application and target**
  - **Different granularities of compilation**
  - **New optimization algorithms**

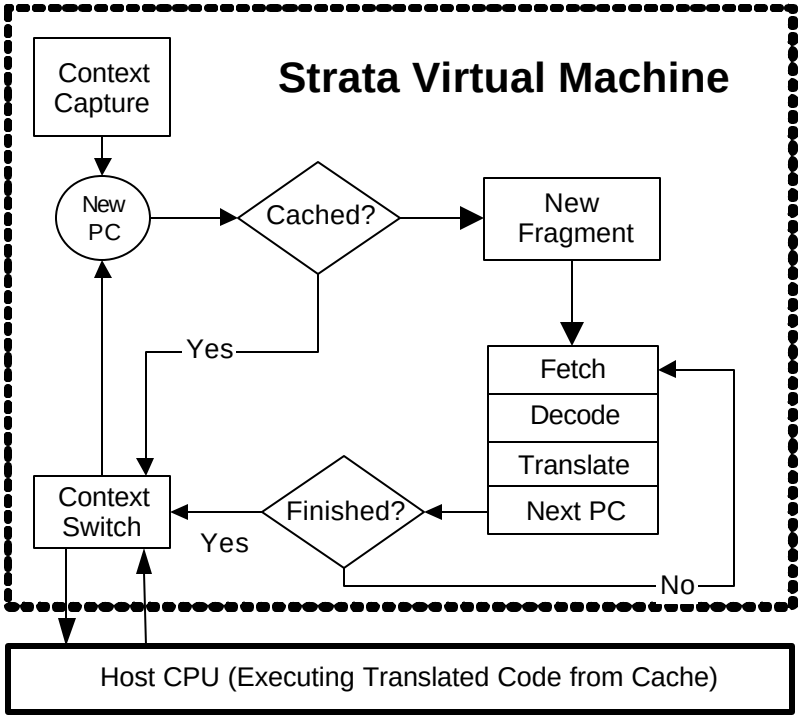
VISTA

- **Framework for building reconfigurable, adaptive compilers**
  - **Optimization backplane for compile-time flexibility**
  - **Language for compile-time configuration of the backplane**
  - **Constraint language for specifying code requirements**
  - **Interactive system for viewing and controlling and understanding optimizer actions**
  - **Varying granularity of optimization (program, function, loop, basic block(s))**

Meeting these challenges

- **Networked computing: software dynamic translation (SDT) (alteration of a running program to achieve some objective)**
  - **Improve performance (Dynamo)**
  - **Overcome economic barriers to hardware innovation (Transmeta)**
  - **Apply application-specific ISA improvements**
  - **Adapt to changes (power, QoS, resource availability)**
  - **Improve security and robustness of code**

Strata: Retargetable SDT Framework



- **Base VM implements a simple SDT providing common services**
- **Programmer implements new SDTs by customizing the VM**
- **VM is customized by overriding functions in the target interface**
- **Currently targeted to SPARC and MIPS. ARM and X86 next**

Research challenges (Partial)

- **Adaptive/reconfigurable compilation**
  - **New optimization approaches/algorithms**
  - **Strategies for automatically configuring the compiler**
  - **Better MDs**
- **Software dynamic translation**
  - **Innovative applications of SDT (security, fault tolerance, correctness, code compression)**
  - **Reducing SDT overhead**
  - **Performance analysis and use**
  - **Hardware support for SDT**
  - **Better binary-level tools**